Notes on the Staking Model (Draft version)

Francisco Lepone
francisco@exact.ly

August,2024

Abstract

These notes introduce the first part of a two-module staking program designed to enhance long-term commitment within decentralized finance (DeFi) protocols. The proposed model integrates a flexible dividend module with an integrated penalty function to manage early withdrawals. This mechanism ensures that participants who maintain their stakes over extended periods are rewarded more effectively, thus aligning staker incentives with the project's sustained success. By focusing on both the equitable allocation of dividends and the deterrence of premature exits, this staking framework aims to reduce the volatility associated with token dumping and contribute to the overall stability and growth of the Ξxactly protocol.

1 Introduction

The staking of tokens has become a critical tool in decentralized finance (DeFi), not only for securing blockchain networks but also for aligning the incentives of participants with the long-term success of a project. Initially, staking mechanisms were primarily employed in Proof-of-Stake (PoS) consensus protocols, where token holders would lock up their assets to participate in network validation processes. This model, as seen in projects like Ethereum 2.0 and Cardano, was designed to ensure network security by rewarding honest validators and penalizing malicious activity.

However, as the DeFi ecosystem has evolved, staking has taken on a broader role, particularly in fostering commitment to a project. In addition to its security function, staking now serves as a mechanism to encourage token holders to remain engaged with a project over the long term, reducing the likelihood of token dumping and promoting price stability. By offering participants a share in the revenues generated by the protocol, staking transforms token holders into active stakeholders who have a vested interest in the project's ongoing success.

This alignment of incentives is crucial in DeFi, where the rapid movement of capital can lead to significant volatility. Protocols have therefore begun to design staking schemes that not only secure the network but also reward long-term commitment. For instance, in liquidity mining models used by protocols

like Uniswap and SushiSwap, participants stake their tokens to provide liquidity, earning a portion of trading fees. This not only incentivizes users to keep their tokens staked but also ties their rewards directly to the performance and growth of the protocol.

Despite the progress made, existing staking schemes often struggle to balance flexibility with the need for sustained participation. Many models impose rigid staking periods or fail to adequately reward users who commit their tokens for longer durations. Moreover, the risk of early withdrawals, which can destabilize a protocol, is not always effectively mitigated.

In response to these challenges, this paper introduces a novel staking model that emphasizes long-term commitment through a flexible dividend module and a sophisticated penalty function for early withdrawals. This model is designed to optimize the distribution of protocol revenues to stakers while ensuring that those who commit to the project over extended periods are appropriately rewarded. By incorporating mechanisms that both encourage and enforce sustained participation, this model aims to enhance the alignment of incentives in DeFi, reducing the potential for token dumping and contributing to the overall stability and success of the protocol.

2 Protocol Fees distribution

As the Protocol collects fees from loans, those fees will be allocated for different purposes. Lets call $\Delta fees_{t,t+1}$ the total amount of fees collected in the time interval between t and $t+1$. A fraction θ of them will be assigned to the staking program, the remaining will be left to the treasury for discretional use.

Inside the staking program, a fraction λ will be assigned to the dividend subprogram, and $(1-\lambda)$ to the liquidity provider staking subprogram. So that:

$$\Delta fees_{t,t+1} = \Delta fees^{treas}_{t,t+1} + \Delta fees^{\text{div}}_{t,t+1} + \Delta fees^{liq}_{t,t+1} \quad (1)$$

$$\Delta fees^{treas}_{t,t+1} = (1-\theta)\Delta fees_{t,t+1} \quad (2)$$

$$\Delta fees^{\text{div}}_{t,t+1} = \theta\lambda\Delta fees_{t,t+1} \quad (3)$$

$$\Delta fees^{liq}_{t,t+1} = \theta(1-\lambda)\Delta fees_{t,t+1} \quad (4)$$

where θ and λ are parameters determined by governance.

3 Dividend Module

The dividend module receives a portion of the treasury fees generated by the protocol and distributes them among stakers, taking into account the duration of their participation in the staking program. There are a number possible rules for dividend distribution based on the holding period. One of the simplest approaches is to define a reference staking period, T_{ref}, and create a corresponding dividend index. If dividends are distributed in multiple assets, a reference index must be created for each one. A simpler approach is to convert all dividends to a single asset and use a unique index.

The index therefore represents the accumulated dividends by unit of staked asset. This index can be updated either at regular predefined intervals or whenever a new transaction occurs within the protocol.

Assuming there is an amount $\Delta fees^{\text{div}}_{t,t+1}$ to be distributed between two consecutive updates, the index will be adjusted as follows.

$$ind_{t+1} = ind_t + \frac{\Delta fees^{\text{div}}_{t,t+1}}{tot\_stk\_asset_t}, \tag{5}$$

where $tot\_stk\_asset_t$ is equal to the total number of staked assets at time t, and $ind_0 = 0$.

We will also need to update:

$$tot\_stk\_asset_{t+1} = tot\_stk\_asset_t \tag{6}$$

The proposed scheme allows users to freely add or withdraw assets from the staking program by modifying the average staking entry time and the average staking entry index level. Each time a user wants to stake at time t_X an amount X of assets, the system performs the following tasks:

- At global staking level

1. - (a) First, update the index with current amount of total assets

$$ind_{t_X} = ind_{t-1} + \frac{\Delta fees^{\text{div}}_{t-1,t_X}}{tot\_stk\_asset_{t-1}} \tag{7}$$

   (b) Then update the total amount of staked assets

$$tot\_stk\_asset_{t_X} = tot\_stk\_asset_{t-1} + X \tag{8}$$

- At the user level

1. - (a) Define the current weight for the user

$$w = \frac{us\_stk\_asset_{prev}}{us\_stk\_asset_{prev} + X} \tag{9}$$

(b) Update and save user´s average staking entry time

$$t_{stk} = w * t_{stk}^{prev} + (1 - w) * t_X \tag{10}$$

(c) Update and save user average staking entry index

$$us\_ind_{t_{stk}} = w * us\_ind_{t_{stk}}^{prev} + (1 - w) * ind_{t_X} \tag{11}$$

(d) Update and save the user´s total staked assets

$$us\_stk\_asset_{t_X} = us\_stk\_asset_{prev} + X \tag{12}$$

the sub/supra index *prev* stands for the variable values before the addition of new assets.

Similarly, each time a user wants to withdraw at time t_Y an amount Y of his staked assets, the system does the following:

- At global staking level

1. (a) Update the index with current amount of total assets

$$ind_{t_Y} = ind_{t-1} + \frac{\Delta fees_{t-1,t_Y}^{\text{div}}}{tot\_stk\_asset_{t-1}} \tag{13}$$

(b) Update the total amount of staked assets

$$tot\_stk\_asset_{t_Y} = tot\_stk\_asset_{t-1} - Y \tag{14}$$

- At the user level

1. (a) Update and save user total staked assets

$$us\_stk\_asset_{t_Y} = us\_stk\_asset_{before} - Y \tag{15}$$

(b) Pay user the amount

$$\max\left(0,\ Y * disc_{fact} * [ind_{t_Y} - us\_ind_{t_{stk}}] - \sum_s prev\_pay(s)\right) \tag{16}$$

where $\sum prev\_pay$ is the sum of all previous payments and, $disc_{fact}$ is a penalty function to be defined in the next section

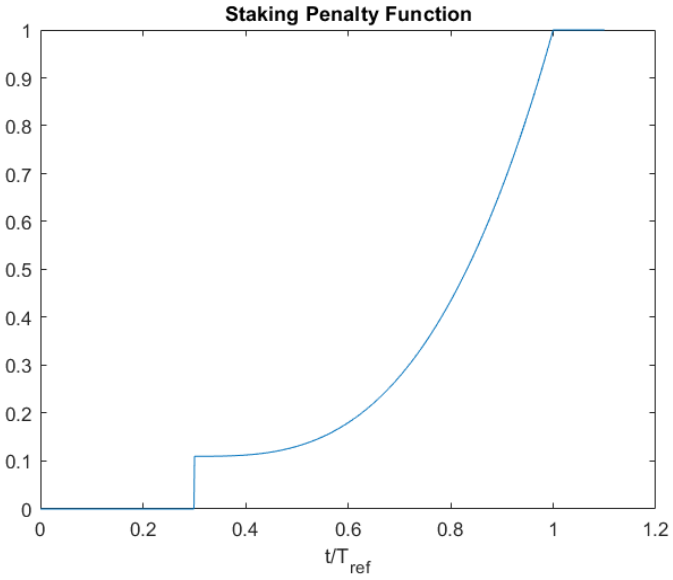


Figure 1: Penalty function behavior.

3.1 Penalty function (discount factor)

The penalty function is designed to discourage early withdrawals from the staking program. This function can be tailored to accommodate both soft and hard-locking periods.

First, we define a reference staking period, τ_{ref}. Staking for a duration shorter than this period will result in a reduction of the dividends assigned to the staker. Additionally, we define a minimum staking period $\tau_{\min}$, where $\tau_{\min} < \tau_{ref}$. Staking for less than $\tau_{\min}$ may either be prohibited (hard-locking) or subject to zero rewards (soft-locking).

Furthermore, we introduce a threshold penalty factor β, which represents the minimum fraction of dividends that a staker is eligible to receive (it could make sense to set $\beta \neq 0$ in the case of hard-locking).

Then, the penalty function can be defined as:

$$disc_{fact}\left(\tau_{stk}\right) = \min\left\{1, (1-\beta) * \left[\max\left(0, \left[\frac{(\tau_{stk} - \tau_{\min})}{(\tau_{ref} - \tau_{\min})}\right]\right)\right]^{\alpha} + \beta\left(\tau_{stk} > \tau_{\min}\right)\right\} \tag{17}$$

where $\tau_{stk} = t - t_{stk}$ is he period of time the assets have been staked, and α controls the growth rate of the discount factor ($disc_{fact}$). Fig.(1) illustrates how $disc_{fact}$ might behave as a function of the staking period (τ_{stk}).

Fig.(2) shows the penalty function as a function of different $T_{\min}$ values.

Fig.(3) shows the penalty function as a function of different α values.

Fig.(4) shows the penalty function as a function of different β values.

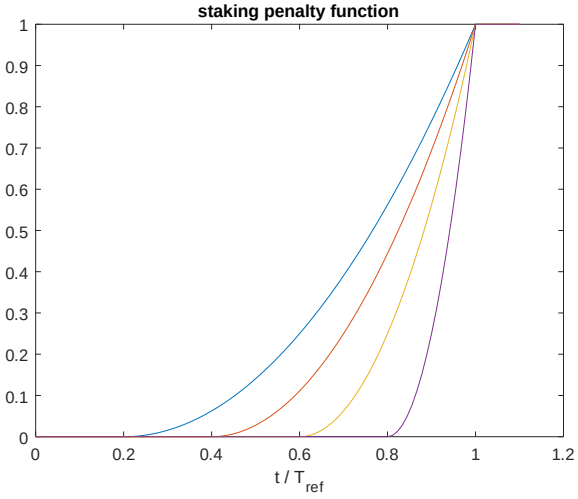


Figure 2: Staking function for $\frac{T_{\min}}{T_{ref}} = 0.2, 0.4, 0.6, 0.8$.

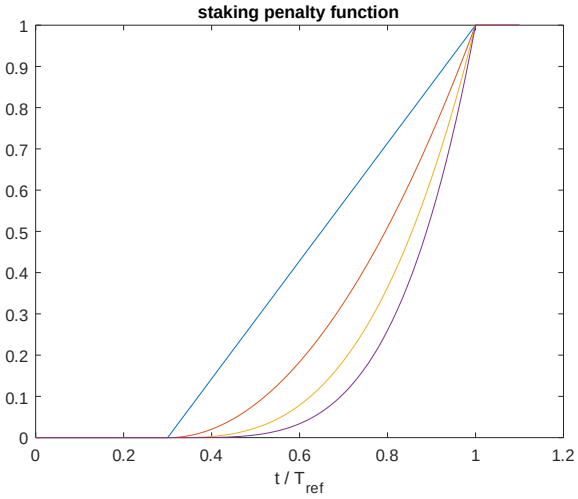


Figure 3: Staking function for $\alpha = 1, 2, 3, 4$.

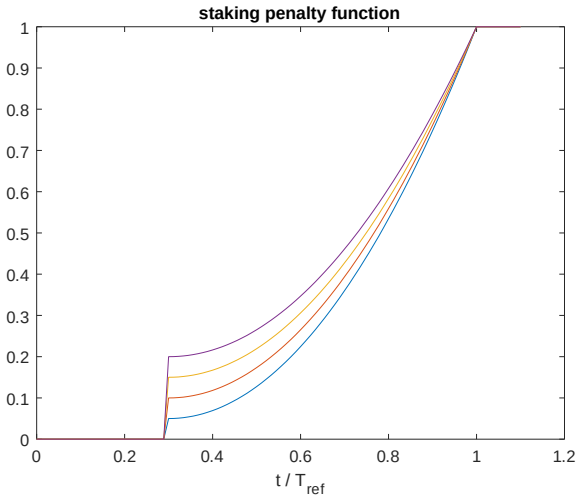


Figure 4: Staking function for $\beta = 5\%, 10\%, 15\%, 20\%$

4 Beyond T_{ref}

In this section, we describe the treatment when the staking period (τ_{stk}) is longer than the maximum reference staking period T_{ref}.

In such a case, the idea is to pay in full for the maximum predetermined period and to apply a discount for any excess over this maximum. Given the characteristics of the penalty function, users won´t be allowed to add assets to an existing staking program if the staking period is greater than the reference period ($\tau_{stk} > T_{ref}$). In such a case, all the staked asset will be unstaked and a new staking program initiated.

Within this framework, the dividend amount for withdrawing Y assets at an horizon $\tau_{stk} = (t - t_{stk}) > T_{ref}$ can be written as):

$$Y * \left(ind_{T_{ref}+t_{stk}} - ind_{t_{stk}}\right) + \gamma * Y * \left(ind_t - ind_{T_{ref}+t_{stk}}\right) \tag{18}$$

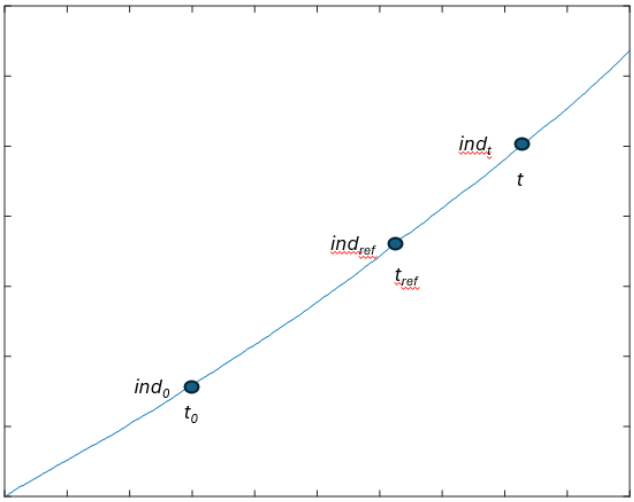
where $ind_{t_{stk}}$ is the initial index at staking (or the equivalent average), and $\gamma, 0 \leq \gamma \leq 1$ is the discount factor for excess exposure. The problem with the expression above is that $ind_{T_{ref}+t_{stk}}$ is unknown at withdrawal time t.

It is necessary to introduce an approximation scheme to infer the value of $ind_{T_{ref}+t_{stk}}$. Choose a linear function defined by:

$$ind_{T_{ref}+t_{stk}} = (ind_t - ind_{t_{stk}}) \frac{T_{ref}}{\tau_{stk}} + ind_{t_{stk}} \tag{19}$$

Note that when $t = T_{ref} + t_{stk}$ then $\tau_{stk} = T_{ref}$ and $ind_{Tref+t_{stk}} = ind_t$.

We can rewrite Eq.(18) as:



$$
\begin{aligned}
& Y * \left[(1-\gamma) * ind_{T_{ref}+t_{stk}} + \gamma * ind_t - ind_{t_{stk}}\right] \\
& Y * \left[(1-\gamma) * \left((ind_t - ind_{t_{stk}})\,\frac{T_{ref}}{\tau_{stk}} + ind_{t_{stk}}\right) + \gamma * ind_t - ind_{t_{stk}}\right] \\
& Y * \left[(1-\gamma) * (ind_t - ind_{t_{stk}})\,\frac{T_{ref}}{\tau_{stk}} + (1-\gamma) * ind_{t_{stk}} + \gamma * ind_t - ind_{t_{stk}}\right] \\
& Y * \left[\left((1-\gamma) * \frac{T_{ref}}{\tau_{stk}} + \gamma\right) * ind_t - \left((1-\gamma) * \frac{T_{ref}}{\tau_{stk}} + \gamma\right) * ind_{t_{stk}}\right]
\end{aligned}
\tag{20}
$$

So that,

$$
Y * \kappa * (ind_t - ind_{t_{stk}})\,, \tag{21}
$$

$$
\kappa \quad = \quad \left((1-\gamma) * \frac{T_{ref}}{\tau_{stk}} + \gamma\right) \tag{22}
$$

Note that when $\tau_{stk} = T_{ref}$, $\kappa = 1$ and we recover the usual expression $(Y * \left[ind_{T_{ref}+t_{stk}} - ind_{t_{stk}}\right])$. It is worth mentioning that κ can be thought as the value of $disc_{fact}$ when $\tau_{stk} = t - t_{stk} > T_{ref}$.

5 Partial dividend withdrawal by stakers and redistribution to the treasury

At any time while staking their assets, stakers can withdraw current accrued dividends. This is possible because accrued dividends are a monotonic increasing function of staking time. The only requisite is to define a variable that keeps track of past withdrawals. So the effective amount of dividends a user can withdraw at any time is given by

$$dvw(\tau_w) = \max\left(0,\ Y * disc_{fact}(\tau_w) * [ind_{\tau_w} - us\_ind_{stk}] - \sum_s dvw(s)\right) \tag{23}$$

where $\sum_s dvw(s)$ is the cumulative value of all past withdrawals. Note that $\sum_s dvw(s) = \sum_s prev\_pay(s)$

Regarding the disposal of dividends not captured by stakers, there are a few consideration to make. First, each time a user unstake their assets the smart contract can collect and redirect unused pending dividends. Second, without the direct action of the wallet owner it is not possible to claim any unused dividend before τ_{ref} since the staker has the chance to claim their dividends in full at a later time by waiting until τ_{ref}. Third, for $\tau_{stk} > \tau_{ref}$, it seems feasible for the staking program to partially withdraw unused dividends based on the following: When $\tau_{stk} > \tau_{ref}$, the staker can claim the amount given by

$$\max\left(0,\ Y * \kappa * (ind_t - ind_{t_{stk}}) - \sum_s prev\_pay(s)\right) \tag{24}$$

with κ given by Eq.(21). The the unclaimable fraction of dividends is given by

$$\max\left[0,\ Y * (ind_t - ind_{t_{stk}}) - \max\left(Y * \kappa * (ind_t - ind_{t_{stk}})\,,\ \sum_s prev\_pay(s)\right)\right], \tag{25}$$

Looking at Eq.(25) we see that, if different from 0, it take two alternative forms. Either, it can be

$$Y * (ind_t - ind_{t_{stk}}) - \sum_s prev\_pay(s) \tag{26}$$

which is always positive. Or, it can be

$$Y * (1 - \kappa) * (ind_t - ind_{t_{stk}}) \tag{27}$$

$$Y * (1 - \gamma) \left(1 - \frac{T_{ref}}{\tau_{stk}}\right) * (ind_t - ind_{t_{stk}}) \tag{28}$$

That ´s basically the product of $\left(1 - \frac{T_{ref}}{\tau_{stk}}\right)$ which is a positive growing function of τ_{stk} (in fact, it´s a function $\in [0, 1]$) and $(ind_t - ind_{t_{stk}})$ which is also a growing function of τ_{stk}. That means that unused dividends always increase with τ_{stk} therefore allowing partial claims and redistributions.

One final comment about where unused dividends should go. There are many ways to allocate them; however, the best choice seems to be reinjecting them in the protocol fees flow so they follow the allocation rules described in the first section of these notes.

6 Fungible asset property

This section contains an evaluation of the implications of staked-asset fungibility before T_{ref}. We already know that if the user hold the assets until the new T_{ref}.(weighted average of all the partial staking) we will receive the same amount of dividends he would have received should each partial staking be treated as an individual one. The question is what happens if he decides to unstake before T_{ref}. Could he get any advantage by deciding to unstake before T_{ref}?

First, let start by evaluating the total claimable dividend amount form two arbitrary staking decission

$$A_1 * K_1 (ind_t - ind_1) + A_2 * K_2 (ind_t - ind_2) \tag{29}$$

where A_1, A_2 are the asset amounts staked, K_1, K_2 the discount factors, and ind_1, ind_2 the index values at entry time, respectively. Rewriting the expression

$$(A_1 * K_1 + A_2 * K_2) * \left[ind_t - \frac{A_1 * K_1 ind_1 + A_2 * K_2 ind_2}{A_1 * K_1 + A_2 * K_2}\right] \tag{30}$$

$$(A_1 + A_2) * \frac{A_1 * K_1 + A_2 * K_2}{A_1 + A_2} * \left[ind_t - \frac{A_1 * K_1 ind_1 + A_2 * K_2 ind_2}{A_1 * K_1 + A_2 * K_2}\right] \tag{31}$$

$$(A_1 + A_2) * \widetilde{K} * \left[ind_t - \widetilde{ind}\right] \tag{32}$$

where $\widetilde{K} = \frac{A_1 * K_1 + A_2 * K_2}{A_1 + A_2}$ and $\widetilde{ind} = \frac{A_1 * K_1 ind_1 + A_2 * K_2 ind_2}{A_1 * K_1 + A_2 * K_2}$.

The fungible asset scheme can only be applied when previous staked assets have been staked for a period $\tau_1 \leq T_{ref}$. For $\tau_1 > T_{ref}$, existing staking will be finished and a new staking program initiated with the new total amount.

In the case $\tau_1 \leq T_{ref}$, the claimable dividend amount is given by

$$(A_1 + A_2) * K(\tau_{avg}) * [ind_t - ind_{avg}] \tag{33}$$

$$\tau_{avg} = \frac{A_1\tau_1 + A_2\tau_2}{A_1 + A_2} \tag{34}$$

$$ind_{avg} = \frac{A_1 ind_1 + A_2 ind_2}{A_1 + A_2} \tag{35}$$

One desirable property the scheme should have is that Eq.(33) will never be greates than Eq.(32).

Provided that $K(\tau)$ is non concave, we can assure that

$$K(\tau_{avg}) \leq \widetilde{K} \tag{36}$$

If we also have that $ind_{avg} > \widetilde{ind}$, both conditions are sufficient to prove the desired property. Lets find under what conditions, it hols true that

$$ind_{avg} \geq \widetilde{ind}$$

$$\frac{A_1 \cdot ind_1 + A_2 \cdot ind_2}{A_1 + A_2} \geq \frac{A_1 \cdot K_1 \cdot ind_1 + A_2 \cdot K_2 \cdot ind_2}{A_1 \cdot K_1 + A_2 \cdot K_2}$$

Since the denominetors are non negative,

$$(A_1 \cdot ind_1 + A_2 \cdot ind_2) \cdot (A_1 \cdot K_1 + A_2 \cdot K_2) \geq (A_1 \cdot K_1 \cdot ind_1 + A_2 \cdot K_2 \cdot ind_2) \cdot (A_1 + A_2)$$

Expanding both sides:

$$A_1^2 \cdot ind_1 \cdot K_1 + A_1 \cdot ind_1 \cdot A_2 \cdot K_2 + A_2 \cdot ind_2 \cdot A_1 \cdot K_1 + A_2^2 \cdot ind_2 \cdot K_2 \geq A_1^2 \cdot K_1 \cdot ind_1 + A_1 \cdot A_2 \cdot K_1 \cdot ind_1 + A_2 \cdot A_1 \cdot K_2 \cdot ind_2 + A_2^2$$

Rearranging similar terms:

$$A_1 \cdot A_2 \cdot ind_1 \cdot K_2 + A_1 \cdot A_2 \cdot ind_2 \cdot K_1 \geq A_1 \cdot A_2 \cdot ind_1 \cdot K_1 + A_1 \cdot A_2 \cdot ind_2 \cdot K_2$$

Simplifying and grouping terms:

$$A_1 \cdot A_2 \cdot (ind_1 \cdot K_2 + ind_2 \cdot K_1) \geq A_1 \cdot A_2 \cdot (ind_1 \cdot K_1 + ind_2 \cdot K_2)$$

Since $A1 > 0$ and $A2 > 0$, we can divide both sides by $A1 \cdot A2$:

$$ind_1 \cdot K_2 + ind_2 \cdot K_1 \geq ind_1 \cdot K_1 + ind_2 \cdot K_2$$

Rearranging terms:

$$ind_1 \cdot K_2 - ind_1 \cdot K_1 + ind_2 \cdot K_1 - ind_2 \cdot K_2 \geq 0$$

Factoring:

$$\begin{aligned} ind_1 \cdot (K_2 - K_1) + ind_2 \cdot (K_1 - K_2) &\geq 0 \\ (K_1 - K_2) \cdot (ind_2 - ind_1) &\geq 0 \end{aligned} \tag{37}$$

Given that $ind2 \geq ind1$ and $K2 \leq K1$, it follows that Eq.(37) is always non negative. Therefore it is always true that

$$(A_1 + A_2) * K(\tau_{avg}) * [ind_t - ind_{avg}] \leq (A_1 + A_2) * \widetilde{K} * \left[ind_t - \widetilde{ind}\right] \tag{38}$$

7 Voting power and $esEXA$ rewards

Staking shares in the protocol provides additional voting power to users. Define vpb_{ref} as the maximum percentage of extra voting power a user can get through staking their assets. Typically, $0 < vpb_{ref} \leq 1$ (although greater than unity values are also acceptable). The voting power per share will be given by the simple expression:

$$1 + vpb_{ref} * \min\left[1, \frac{\tau_{stk}}{T_{ref}}\right] \tag{39}$$

where by definition $t \geq t_{stake}$.

The same approach can be adopted to provide an additional dividend in terms of $esEXA$. Defining the maximum dividend rate as r_{esExa}, we have

$$disc_{fact} * r_{esExa} \min\left[1, \frac{\tau_{stk}}{T_{ref}}\right] \tag{40}$$

8 Appendix:

8.1 Parameter definitions and range

- $T_{\min} \rightarrow$ Hard lock period $\in [0, T_{ref}]$
- $T_{ref} \rightarrow$ Reference staking period $T_{ref} > 0$ (typically, $3M, 6M, 1Y$)
- $\beta \rightarrow$ Minimum reward fraction $\in [0, 1]$
- $\alpha \rightarrow$ Discount factor growth exponent $\in [0.1, 100]$
- $\gamma \rightarrow$ Discount factor for excess exposure $\in [0, 1]$

8.2 Scaling assets

In order to avoid rounding problems originated when doing multi-staking of different asset sizes, we propose the following scale transformation. Let start by ans scheme where no penalty factor applies. The dividend amount of a two different staking position is given by:

$$A_1 \ (ind_t - ind_1) + A_2 \ (ind_t - ind_2) \tag{41}$$

Let´s define

$$A_{tot} = A_1 + A_2 \tag{42}$$

$$A_1^* = \frac{A_1}{A_{tot}} \tag{43}$$

$$A_2^* = \frac{A_2}{A_{tot}} \tag{44}$$

Then

$$A_1 \ (ind_t - ind_1) + A_2 \ (ind_t - ind_2)$$

$$(A_1 + A_2) \ ind_t - (A_1 \ ind_1 + A_2 \ ind_2)$$

$$A_{tot} \ \left[ind_t - \left(A_1^* \ ind_1 + A_2^+ \ ind_2\right)\right]$$

$$A_{tot} \ (ind_t - ind_{avg})$$

where

$$ind_{avg} = A_1^* \ ind_1 + A_2^+ \ ind_2 \tag{45}$$